Georgia Technology Authority

Roy E. Barnes, Governor

Larry J. Singer, Chief Information Officer, State of Georgia
and Executive Director, Georgia Technology Authority

Software Reuse Vision

Version 1.2

Revision History

Date	Version	Description	Author
<dd/mmm/yy>	<x.x>	<details>	<name>
18/Mar/02	1.0	Initial Strategic Vision	S.L.Clarke
29/Mar/02	1.1	e-Strategy Review	S.L.Clarke
02/Apr/02	1.2	NCSE – update	S.L.Clarke

Table of Contents

1. Executive Summary 1
 1.1 Purpose 1
 1.2 Scope 1
 1.3 Definitions, Acronyms and Abbreviations 1
 1.4 Vision Statement 1
 1.5 Overview 1

2. Positioning 3
 2.1 Business Opportunity 3
 2.2 Problem Statement 5
 2.3 Program Position Statement 5
 2.4 Reuse and CMM 6
 2.5 Reuse and the national software component exchange 7

3. Stakeholder and User Descriptions 8
 3.1 Stakeholder Summary 8
 3.2 Stakeholder Profiles 9
  3.2.1 Producer 9
  3.2.2 Consumer 10
  3.2.3 Broker 11
 3.3 User Summary 12
 3.4 User environment 15
 3.5 Key Stakeholder / User Needs 15

4. Overview of reuse adoption strategy 16
 4.1 Initiating 16
 4.2 Investigation 16
 4.3 Planning 19
 4.4 Implementation – reuse pilot program 20
 4.5 Continuous Improvement 21
 4.6 Assumptions and Dependencies 22
 4.7 Time and Resources 22

5. Program Features 24
 5.1 Re-use Artifact Life Cycle 24
 5.2 Vision statement 24
 5.3 Implementation strategy 24
 5.4 Reuse organizational structure 24
 5.5 Reuse roles and responsibilities 25
 5.6 Reuse metrics 25
 5.7 Reuse marketing strategy 25
 5.8 Reuse education and training plan 25
 5.9 Pilot project 25

6. Constraints 26

7. Other Program Requirements 26
 7.1 Applicable Standards 26

7.2 System Requirements 26

Appendix A Reuse Definitions 27

Appendix B Bibliography 32

State of Georgia's Vision for Software Reuse

1. Executive Summary

The purpose of this document is to collect, analyze and define high-level needs and features of a software reuse strategy. It focuses on what will be needed to be successful, who the stakeholders in a reuse initiative are, the target users, and why these needs exist.

1.1 Purpose

The software reuse vision and mission statements describe the Georgia Technology Authority (GTA) and the State of Georgia's desire to provide a common goal for all of the producers and consumers of software products within the State Government. This document describes this vision and provides a preliminary mission statement for each of the core participants. Conceiving an appropriate vision and mission statements is an iterative process; this is a living document; as the GTA moves forward adopting a software reuse strategy these initial statements will be revisited and refined as necessary.

1.2 Scope

To specify the processes, activities and tasks for extending the software life cycle process to include the systematic practice of software re-use. The scope of these processes are broader than a single project since reuse processes such as "Domain Engineering" transcend the life cycle of any single project and apply to a set of related software products.

1.3 Definitions, Acronyms and Abbreviations

See Appendix A

1.4 Vision Statement

The vision of the GTA software reuse initiative is to drive the State of Georgia's software community from its current 're-invent the software' cycle to a process-driven domain-specific architecture-centric, library-based way of constructing software. The proposed strategy to realize this vision is based on systematic reuse: where opportunities are predefined and a process for capitalizing on those opportunities is realized.

1.5 Overview

The goal of the State of Georgia's software reuse program is to improve the efficiency of software development by 20 percent and software maintenance by 40%.

To achieve some of these goals it is necessary for the State to plan ahead to maximize systematic re-use, maximize the potential return on investment, and minimize risks associated with adopting a new software engineering process.

The benefits from reuse will not be recuperated during a reuse pilot but accumulated over the life of the reuse initiative.

The necessity to develop and roll out the reuse program in step with increasing the maturity level of the software development teams cannot be overemphasized. The project team should preferably already be proficient in component based development (CBD) and have a number of successful CBD projects to its credit.

The number of people required to implement a reuse program will fluctuate during the life cycle of the reuse program. The impact of the reuse program is proportional to the size of the reuse team and the size of the organization it is trying to influence.

The GTA should not undertake a review of reuse potential at this time, but instead applies industry metrics based on the State of Georgia's software development and maintenance budgets.

The GTA should ensure that targeted software development communities within the State of Georgia have achieved a minimum level of capability as a parallel but critical goal for a successful reuse program. It is recommended that a development plan to meet reuse aptitude is developed during investigation and implemented in parallel with the planning phase.

The planning phase is critical to the success of a systemic reuse program. During the planning phase one or more projects will be selected. These projects will serve as a test for the proposed reuse practices and a showcase for wider deployment.

The GTA should adopt an incremental pilot implementation plan, developed during the planning phase. This will ensure that the existing scarce human resources are not over stretched during implementation.

The GTA does not currently have available resources with the skills that will be required during the planning phase.

To accelerate the reuse program during the implementation phase, the GTA should partner with external development contractors who have existing reuse and CMM[1] level 5 component based development programs.

The implementation or "technology transfer" of reuse is a critical aspect to achieve systemic reuse. A reuse program planned down to the smallest detail is of no value if this technology cannot be successfully transferred to the agencies that would most benefit.

It's important to establish specific goals for the level of reuse, these goals must be both reasonable and realistic. The reuse goals of the GTA will identify the appropriate critical set of metrics for benchmarks, incentives and decision making.

A reuse marketing strategy will be developed this may include reuse visionaries, reuse library prototypes, reuse educators, communication plans, product pricing, product launch events, distribution and promotional activities.

Selecting and initiating a reuse pilot will need to be co-coordinated with the State Funding Cycle to ensure that resources are available at the end of each phase to implement the subsequent activities.

This program will complement the component repository initiative. The reuse team created to support this project will be able to ensure that quality components are commissioned and acquired by the State of Georgia.

All components created for the State will be constructed to conform to the *Reusable Asset Specification.*

The remainder of this vision document describes the business opportunity that implementing a reuse strategy can yield. Identifies alternatives to implementing a systemic reuse strategy, identifies the key constituents, stakeholders and users of a systemic reuse program and provides an overview of the recommended steps to successful adopt and execute a reuse strategy with a breakdown of the artifacts that will need to be created during the reuse planning phase.

[1] Capability Maturity Model

2. Positioning

2.1 Business Opportunity

The success and survival of enterprises in 2002 and beyond will require that application solutions can be rapidly deployed and reassembled targeting a variety of software platforms. To achieve this level of speed and flexibility, organizations must adopt a service-oriented architecture based on reusable components involving all software assets - including packages, legacy applications and new component-based services. To maximize the return on investment and minimize the risks in making such a fundamental change, enterprises must implement a formal reuse program, including a component catalog and related infrastructure and methodological changes in support of the reuse objectives, as part of its long-term strategies.

Michael Blechar, vice president of Internet and e-Business Technologies at Gartner, Inc.

To achieve some of these goals it is necessary for the State to plan ahead to maximize systematic re-use, maximize the potential return on investment, and minimize risks associated with adopting a new software engineering process.

Benefits of implementing software reuse program

The benefit and costs of implementing a software reuse program are both tangible and intangible in nature.

The major recurrent costs associated with implementing reuse are higher development costs, development costs are typically anywhere from a factor of 1.5 to 2.0 higher than in an organization that is not implementing a reuse strategy.

Savings from reuse are achieved in two of the software lifecycle disciplines: development and maintenance. Savings in development are achieved when existing reusable components are used to assemble new software. Savings in maintenance occur due to reductions in original code and the higher quality associated with a reusable component that has been tested and deployed multiple times. Savings in maintenance are generally of greater value than savings achieved in the development cycle since the maintenance lifecycle typically accounts for 60-80% of the total project cost.

The following chart provides a list of some of the tangible cost and savings from implementing reuse.

Costs of making software reusable	**Savings from Re-use**
25% Generalization	black box[2]
15% Documentation	80% Reduction in development
15% Testing	160% Reduction in maintenance
5% Support & Maintenance	white box
	20% Reduction in Development
Total Cost	**Total Saving**
60% Additional cost	20% - 240% Saving

Dr. Martin Griss, co-author of Software Reuse suggests a similar albeit more reserved projection for white box reuse. "Sometimes, when pushed, I will say ten to fifteen percent."

[2] See Appendix A References

For a reuse pilot project, consultant William Council, co-author of Component-Based Software Engineering, agrees with Dr. Griss. "No more than ten to fifteen percent of software component assets will be used. In many settings, I cannot imagine that software component reuse would reach even ten percent."

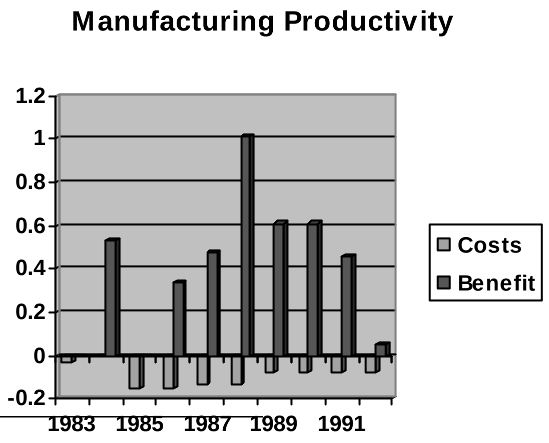
As can be seen from a cost based perspective, black box reuse should be pursued rather than white box reuse whenever the opportunity arises. This should not however deter the State of Georgia from pursuing component based development. Even with no reuse the following benefits typically may be accrued:

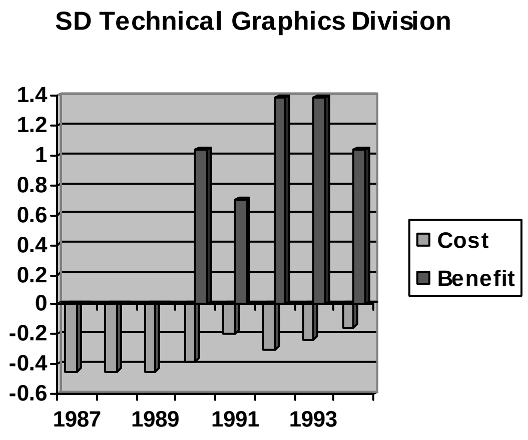
- Building in components makes application maintenance easier and cheaper
- There is less need for expensive developers. You need staff to design applications and assemble components.
- A component can be removed from an application and replaced with a new component, without impacting the entire application. This also reduces the time needed for regression testing.

Economic savings from reuse

The following table summarizes economic information from two reuse programs at HP. Their respective net present values cannot be compared because the business benefits are calculated over two different time lines ten and eight years respectively.

Organization	Manufacturing Productivity Division	SD Technical Graphics Division
Time Line	Ten Years (1983-1992)	Eight Years (1987-1994)
Startup Resources	26 Person Months $0.3M	107 Person Months $1.3M[3]
Ongoing Resources	54 Person Months $0.7M	99 Person Months $1.3M
Gross Cost	80 Person Months $1.0M	206 Person Months $2.6M
Gross Savings	328 Person Months $4.1M	446 Person Months $5.6M
ROI	310%	115%
NPV	125 Person Months $1.6M	75 Person Months $0.9M
Break-even	Year 2	Year 6





[3] $2.0M in yr 2002 dollars note year 1994 months are more expensive than 1983

Note in each case the benefits from reuse were not recuperated during the reuse pilot but accumulated over the life of the reuse initiative. In the second case no tangible benefits occurred until the fourth year of the reuse program, and break even (cumulative benefit > cumulative cost) did not occur until year six.

2.2 Problem Statement

The problem of	Expensive to maintain error prone software systems, high risk software projects and short time to market cycles. Large degree of internal redundancy.
affects	Georgia Constituents, State Agencies, the GTA.
The impact of which is	Reduced service and increased cost of service.
A successful solution would	Reduce overall cost of ownership, improve reliability, improve software quality, increase software development productivity, decrease length of development cycles, and increase constituent satisfaction.

2.3 Program Position Statement

For	Georgia Technology Authority, All agencies except those under the authority, direction , or control of the General Assembly or state-wide elected officials other than the Governor
Who	Require a set of engineering principals to help the organization achieve greater efficiency.
The software reuse strategy	is a strategic initiative.
That	Systematically outlines the resources, personnel, activities and desired outcomes for implementing a reuse program.
Unlike	Unified Process or CMM.[4]
Software Reuse	Is not a software engineering process. It is designed to complement and extend existing best practices not replace them. (Ref Section 2.4 Reuse and CMM)

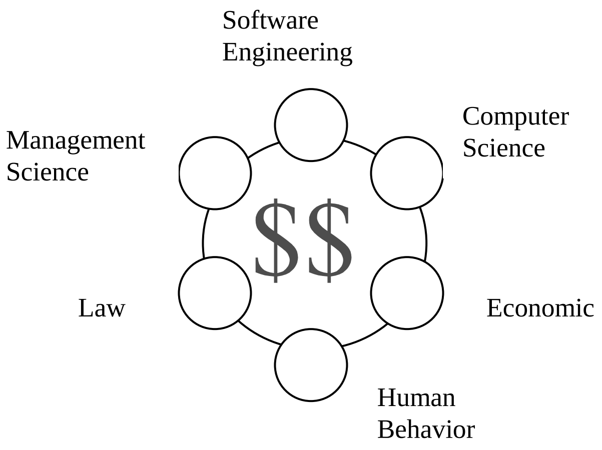
The software reuse program for the State of Georgia will provide a multidisciplinary approach to software reuse; software reuse not limited to “software components” but addressing all aspects and assets created during the software engineering lifecycle. These include:

- Architecture
- Source Code
- Data
- Designs
- Documentation

[4] Capability Maturity Model

- Estimates
- Human Interfaces
- Plans
- Requirements
- Test Cases

A Multidisciplinary Approach



Developing a reuse adoption strategy is the art of generating and employing resources for the formal acceptance and institutionalization of software reuse followed by the diffusion of innovation. Participation from representatives in each of the disciplines identified in the preceding diagram will be required during the investigation, planning and implementation phases of the reuse strategy.

2.4 Reuse and CMM

The necessity to develop and roll out the reuse program in step with increasing the maturity level of the software development teams cannot be overemphasized. The pilot project team should preferably already be proficient in component based development (CBD) and have a number of successful CBD projects to its credit.

To accelerate the reuse program in the implementation phase, GTA should partner with external development contractors who have existing reuse and CMM level 5 component based development programs.

A reuse maturity model is a set of stages through which an organization progresses. A maturity model specifies progressively higher levels of capability to which an organization could aspire. Researchers and practitioners have not agreed on the relationship between CMM and reuse. CMM (version 2) contains an organizational software asset key process area at Level 4. Some research has concluded that reuse at lower CMM levels may not result in any net benefit.

e.g. If the GTA were to practice reuse at lower levels with for example defect prone software quality assurance practices this will propagate through to defective assets in an asset repository. While some benefits may be experienced these will however be offset by the likelihood that the process is unsystematic, fixes are not undertaken or are poorly implemented.

CMM maturity framework definitions:

Level 1

Ad hoc, little formal structure, methods and tools not integrated

Level 2

Process appears to be under control but unable to meet evolving needs

Level 3

Strong software engineering base exists, Few qualitative productivity metrics, qualitative success.

Level 4

Development products are scrutinized , formal quality control function implemented

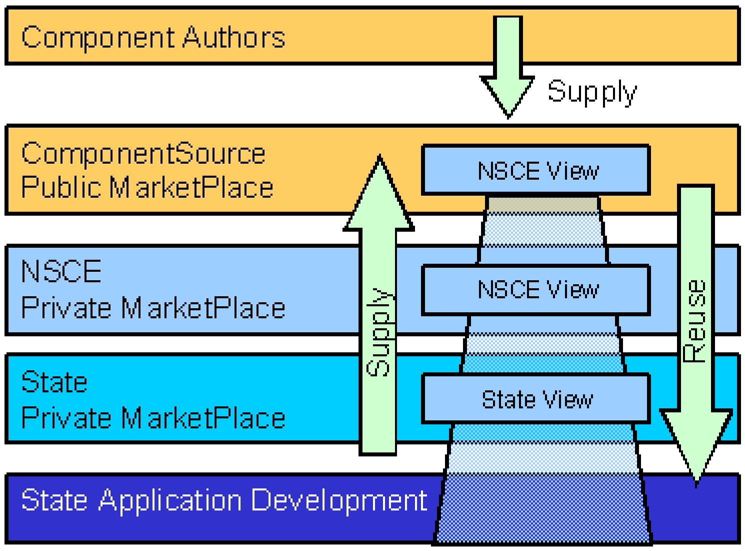
Level 5

Development process scrutinized, statistical quality control, continuous quality improvement.

2.5 Reuse and the national software component exchange

The National Software Component Exchange (NSCE) is a proposal for a national component market place. The National Software Component Exchange would be an online, nationwide private component exchange where states and local governments are able to share knowledge and intellectual property by reusing software components they have built and ease the sharing of data across common applications within and across the different states.

The National Software Component Exchange would consist of three separate repositories, or views, that each state could access:



NASCIO[5] recognized many of the fundamental obstacles in successfully implementing an exchange: Maturity of software providers, consistent documentation, incentives for contribution, intellectual property rights. Each individual state was charged with the responsibility for managing its own exchange. NCSE does not specify the processes, activities and tasks for extending the software life cycle process to include the systematic practice of software re-use. These processes and a plan for the SOG to actively contribute to this process are addressed in this document.

[5] The National Association of State Chief Information Officers

3. Stakeholder and User Descriptions

To effectively provide services that meet your stakeholders' and users' real needs, it is necessary to identify and involve all of the stakeholders as part of the initiation, investigation and planning process. GTA must identify the users of the process and ensure that the stakeholder community represents them adequately.

3.1 Stakeholder Summary

The audience for the GTA software reuse initiative is the producers, brokers and consumers of software products within State Government. All agencies except those under the authority, direction, or control of the General Assembly or state-wide elected officials other than the Governor who are interested in creating, supporting and/or using internal or external markets for reusable software assets. This includes external vendors who have been commissioned by the State of Georgia to produce software assets.

Name	Represents	Role
Name the stakeholder type.	Briefly describe who they are represented by with respect to the initiative.	Briefly describe the role they are playing in the initiative. i.e. Ensure this….
Producer	GeorgiaNet, e-development, vendors, fixed bid projects	Ensure creation of components and other software engineering artifacts developed with latent reuse potential.
Consumer	Georgia Agency Software Developers. GTA e-development	Agency business owner charged with responsibility of providing a software solution and sharing a common goal to reuse pre-existing artifacts to improve quality and decrease development cycles.
Broker	NASCIO, ComponentSource, GeorgiaNet e-development	Provide a marketplace for exchange of software artifacts. Ensure quality and security of software components. Measure reuse uptake.

3.2 Stakeholder Profiles

Representative	*Who is the stakeholder representative to the project (optional - if documented elsewhere). What we want here is names!*
Description	*Brief description of the stakeholder type*
Type	*Qualify the expertise of the stakeholder i.e. GURU, BUSINESS EXPERT, CASUAL USER etci.e. Technical background and degree of sophistication*
Responsibilities	*List the key responsibilities of the stakeholder with regards to the system being developed (i.e. their interest as a stakeholder).*
Success Criteria	*How does the stakeholder define success? How is the stakeholder rewarded?*
Involvement	*How the stakeholder is involved in the project - relate where possible to RUP workers (i.e. Requirements Reviewer etc.)*
Deliverables	*Any additional deliverables required by the stakeholder. These could be project deliverables or output from the system under development.*
Comments / Issues	*Problems that interfere with success and any other relevant information*

3.2.1 Producer

Representative	Brian Copeland
Description	Producer
Type	BUSINESS EXPERT
Responsibilities	Ensure creation of components and other software engineering artifacts developed with latent reuse potential.
Success Criteria	How does the stakeholder define success? How is the stakeholder rewarded?
Involvement	Represents the following RUP business workers • Asset Developer • Asset Harvester • Asset Packager • Artifact Sanitizer
Deliverables	Deliverables required by the stakeholder. • Implementation Strategy • Definition of Organizational Structure • Roles and responsibilities • Education and Training plan • Pilot project
Comments / Issues	Problems that interfere with success and any other relevant information

3.2.2 Consumer

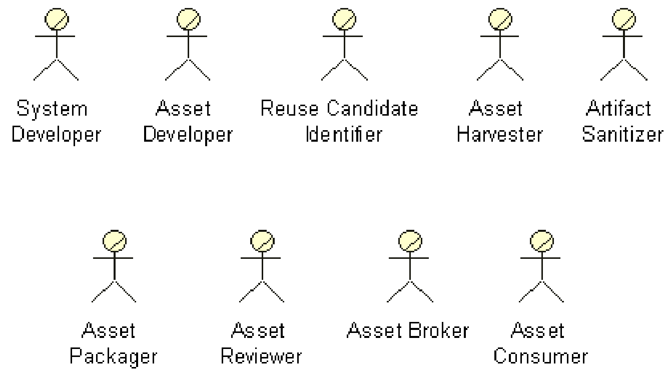
Representative	Jeffrey Huffman, Kerry Bass
Description	Consumer
Type	CASUAL USER
Responsibilities	Agency business owner or GTA e-Development charged with responsibility of providing a software solution and sharing a common goal to reuse pre-existing artifacts to improve quality and decrease development cycles.
Success Criteria	How does the stakeholder define success? How is the stakeholder rewarded?
Involvement	Represents the following RUP business workers • System Developer • Asset Consumer
Deliverables	Deliverables required by the stakeholder. • Implementation Strategy • Roles and responsibilities • A marketing Strategy • An Education and Training plan • Identify pilot project
Comments / Issues	Problems that interfere with success and any other relevant information

3.2.3 Broker

Representative	Robert Woodruf, Larry Singer, Steve Oldham (Component Source)
Description	Broker
Type	BUSINESS EXPERT
Responsibilities	Provide a marketplace for exchange of software artifacts. Ensure quality and security of software components. Measure reuse uptake.
Success Criteria	How does the stakeholder define success? How is the stakeholder rewarded?
Involvement	Represents the following RUP business workers • Asset Developer • Asset Harvester • Artifact Sanitizer • Reuse Candidate Identifier • Asset Packager • Asset Reviewer • Asset Broker
Deliverables	Deliverables required by the stakeholder. • Implementation Strategy • Definition of Organizational Structure • Roles and responsibilities • Development of Metrics
Comments / Issues	Problems that interfere with success and any other relevant information

3.3 User Summary

There are several major roles in the asset creation and reuse process some of these are identified below:



As the State of Georgia's software reuse capabilities mature these roles may be fulfilled by different combinations of internal State of Georgia resources, outsourced resources and vendors. For example, during early adoption the resources to fulfill these may be provided using the following combination:

Early Adoption	
Internal	External
• Reuse Candidate Identifier • Asset reviewer • Asset Consumer	• System Developer • Asset Developer • Asset Harvester • Artifact Sanitizer • Asset Packager • Asset Broker • Asset Consumer
Systemic Reuse Steady State	
• System Developer • Reuse Candidate Identifier • Asset Developer • Asset Harvester • Asset reviewer • Asset Consumer	• Artifact Sanitizer • Asset Packager • Asset Broker • Asset Consumer

Once steady state systemic reuse begins to diffuse throughout the State, all of these roles may eventually be fulfilled by State resources.

A summary list of all the identified user

Name	Description	Stakeholder
Name the user type	Briefly describe what they represent .	List how the user is represented by the stakeholders. i.e. Represented by Stakeholder1
System Developer	The individual or team that builds a software system. The primary intent of the System Developer is to meet certain stakeholder needs and **not necessarily reuse**. The artifacts created by the System Developer are candidates for participating in an asset.	Consumer
Asset Developer	The individual or team responsible for developing an asset organically. This role participates in planned reuse development whereas the System Developer does not.	Producer / Broker
Reuse Candidate Identifier	The individual or team that **tags artifacts as potential reuse candidates**. This is typically a team lead, a design reviewer, an architect, or some senior individual that attempts to determine if an artifact has reuse potential.	Broker
Asset Harvester	The individual or team that harvests from existing systems, artifacts, components, and assets that make up a reusable asset. This is a highly skilled role, typically performed by an **experienced system architect**. This actor identifies repeatable problems, their solutions, and can determine the boundaries and scope of the asset by pointing to candidate artifacts.	Producer/Broker
Artifact Sanitizer	This is a skilled craftsperson (skilled developer) that can take an artifact and under the guidance of the harvester create new artifacts from the original artifacts and prepare them for reuse. As part of this activity this person also performs artifact parameterization - introducing variables and customization points within the artifacts.	Producer/Broker

Asset Packager	The Asset Packager collects the artifacts that have been developed and prepared and packages them in an asset. Typically, this person has some technical writing skills and is motivated by the quality of the packaging, as opposed to the Asset Developer who is motivated by the completion of software artifacts for project milestones. The Asset Packager may write many of the artifacts in the asset's Usage section, such as the asset overview, and so on. The Asset Packager takes the system artifacts (from the Asset Developer and cleans them up - typically giving them a new identity, while leaving the original artifacts in the context of the Asset Developer.	Producer/Broker
Asset Reviewer	The Asset Reviewer verifies and validates an asset for downstream consumption. The reviewer affects the state of the asset, permitting or prohibiting it from being available for reuse.	Broker
Asset Broker	This is the organization that hosts the asset. The catalog may be hosted by the broker as well. The broker may have a catalog that describes the asset.	Broker
Asset Consumer	The Asset Consumer applies and reuses the asset. At the very least, the consumer may search for and review the asset. The Asset Consumer is generally most interested in understanding the variability points of the asset and how it can and should be applied.	Consumer

3.4 User environment

The number of people required to implement a reuse program fluctuates during the life cycle of the reuse program. The impact of the reuse program is proportional to the size of the reuse team and the size of the organization it is trying to influence.

In the economic savings example (Section 2.1 Economic savings from reuse), HP's SD graphics division invested three full time resources during the program startup and two resources once steady state was achieved. This investment enabled them to save \$5.6M over eight years on a \$4M+ / year software development budget.

To have a similar impact on the State of Georgia's development budget, assuming a similarly proportioned team, the team would exceed 300 full time equivalents.

Assumption per year SOG software development budget (Development & maintenance) = \$400 M / year

$$\frac{400 \times 3\ \text{FTE}}{4} = 300\ \text{FTE}$$

The typical, and there is no "typical" reuse program, runs from eight to ten years with the initial investigation, planning and pilot phases occurring over a two to four year time frame.

The reuse program is typically dependant and needs to integrate with the existing software engineering process.

3.5 Key Stakeholder / User Needs

Need	Priority	Concerns	Current Solution	Proposed Solutions
Reduce expensive Software Systems	Medium		System based bid management (ART)	ART and Component Development
Lower error prone Software	Low		External Vendor Competence (CHANCE)	Pre-built Components Modular Design Reuse of tested artifacts
Manage risky project	Low		Traffic	Traffic and reduced development
Decrease time to market	Medium		Increase resources, reduce contingency, increase risk.	Create from pre-built components reducing total development effort
Reduce redundant systems	Medium		Strategic Planning	Strategic Planning and commissioning common component development.
Lower overall cost of ownership	High		Outsource	Reduced maintenance and development costs
Increase constituent satisfaction	High		None	Lower costs, increased service.

4. Overview of reuse adoption strategy

This section provides a high level view of the reuse adoption strategy. This section consists of five subsections, as follows:

- Initiating
- Investigation
- Planning
- Implementation
- Continuous improvement

A number of alternative implementation strategies were evaluated, including Prieto-Diaz[6], Calderia[7] and Whittle, Lam Kelly[8]. While there is no single correct way to implement a reuse program, the majority of these implementation strategies take a similar approach with minor variations in the individual steps and assumptions. For example, the Prieto-Diaz adoption strategy makes implicit assumptions that minimum levels of commitment exist and extends the initiation phase to analyze existing software for potential reuse. Whittle Lam and Kelly take an incremental approach; developing and linking together multiple reuse strategies.

4.1 Initiating

The initiation phase of a reuse implementation consists of a number of activities, education, gathering basic information; understanding the benefits of reuse and where it is in use; identifying sponsors for reuse within the State; and gaining commitment to allocate the resources to move to the investigation phase.

Much of the work associated with this phase of the reuse adoption strategy has already been undertaken, executive sponsorship has been identified; all that remains is to identify and allocate the resources needed to effectively implement the investigation phase.

Resources Reuse champion

♦ **Milestone** Investigation resources allocated

4.2 Investigation

The investigation phase consists of a systematic investigation of the feasibility of software reuse (a feasibility study) with the intent of identifying and determining the potential for reuse within State Government. This revolves around gaining a clear understanding of the business goals. All supporting reuse characteristics should be considered: baseline measurements, infrastructure availability, assets, personnel and technology.

For the GTA to successfully implement a reuse program it must possess/develop two key characteristics:

- Re-use potential *Latent redundancies and opportunities across domains*
- Re-use aptitude *Capacity to mine the re-use potential*

[6] R.Prieto-Diaz "Making software reuse work :an implementation model"

[7] G.Caldiera "Domain Factory and Software Reusability"

[8] B.Whittle,W.Lam, and T.Kelly "A pragmatic approach to reuse introduction in an industrial setting"

Reuse Potential and Aptitude Framework

		Reuse Aptitude	
		No	Yes
Reuse Potential	Yes	Failure Quadrant	Success Quadrant
	No	Failure Quadrant	Failure Quadrant

Reuse potential

A number of characteristics are indicative of reuse potential. These include:

A stable domain (business areas); Promotes the understanding as well as the availability of artifacts for domain analysis. If parts of the domain are unstable this can be an incentive to productize the portion that is relatively stable.

Relaxed performance and hardware restraints; Reusability does not necessarily mean a degradation in system performance, often if a component is optimized for reuse the additional scrutiny can result in faster components however typically there is a trade off between reusability and performance.

High Internal Redundancy; The greater the redundancy within an individual system the greater the opportunity for reuse.

Multiple Iterations; A high number of successive releases can be indicative of “carry-over reuse”.

Reuse management best practices recommend undertaking a review of the reuse potential within an organization as part of a formal reuse adoption strategy; some of the benefits cited are greater organizational awareness and substantially stronger supporting business case. A review of reuse potential will quantify the expected levels of black box vs white box re-use and can be used to determine the relative cost of writing for reuse in comparison to the potential savings.

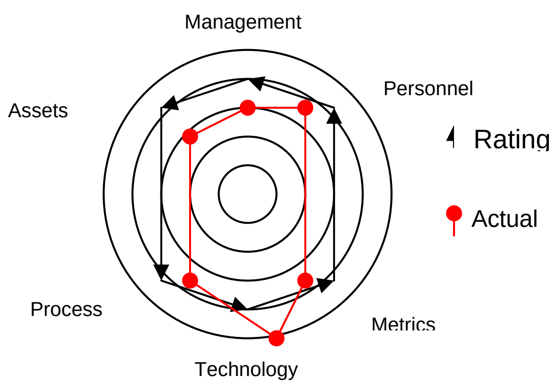
The GTA should not undertake a review of reuse potential at this time, but instead applies industry metrics based on the State of Georgia’s software development and maintenance budgets.

Reuse aptitude

Reuse aptitude is the ability or capacity of the GTA and State agencies to exploit the innate reuse potential. It is recommended that the GTA undertakes an organizational re-engineering for reuse assessment (ORRA) during either the investigative or planning phase. ORRA is a diagnostic method for collecting qualitative and quantitative data on software development with reusable assets. It provides a benchmark for the GTA in six areas:

- Management
- Personnel
- Economics & Metrics
- Technology
- Process
- Reusable assets and products

Sample ORRA Metric



An alternative to undertaking a reuse assessment is to make the pilot implementation dependant on an external milestone such as achieving CMM level 3 accreditation (Ref 2.4 Reuse and CMM).

Accelerated reuse adoption recommendation:

The GTA should ensure that targeted software development communities within the State of Georgia have achieved a minimum level of capability as a parallel but critical goal for a successful reuse program.

Resources		
	Reuse Champion	Communication
	Marketing	Communication
	Managers	Assess re-use potential
	Engineers	Assess re-use potential
	Metrician	Assess re-use aptitude
	Domain Expert	Assess re-use potential

♦ **Milestone** Pilot planning and Pilot resources allocated, GTA meets reuse potential and reuse aptitude checkpoints. Note: it is recommended that a development plan to meet reuse aptitude is developed during investigation and implemented in parallel with the planning phase.

Selecting and initiating a reuse pilot will need to be co-coordinated with the State Funding Cycle to ensure that resources are available at the end of each phase to implement the subsequent activities.

4.3 Planning

The planning phase is critical to the success of a systemic reuse program. During the planning phase one or more pilot projects will be selected. The pilot project will serve as a test for the proposed reuse practices and a showcase for wider deployment. The pilot will also determine the scope and extant of allocated resources.

The activities that occur during the planning phases are:

- Creation of vision statement (This document)
- Implementation Strategy
- Definition of Organizational Structure
- Roles and responsibilities
- Development of Metrics
- Creation of a marketing Strategy
- Creation of an Education and Training plan
- Identify pilot project

Selecting a pilot project(s)

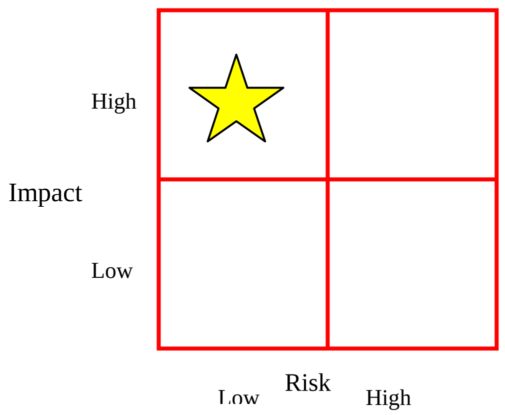
Two approaches for selecting a pilot project have been considered. The first of these is soliciting requests from the agencies to participate in a candidate pilot program; the second is to identify potential projects from existing candidate projects. Both these approaches have a similar purpose in that the project sponsor and the GTA must share a common goal i.e. the success of reuse.

The GTA has additional goals from the pilot project and will need to dedicate additional resources to achieve these objectives, namely capturing information and the methodology for future proliferation. The digital academy may provide a suitable vehicle for achieving these goals.

Pilot Criteria

During the pilot selection phase it will be necessary for the GTA to consider both similarity and diversity. The greater the similarity between the business objectives of the pilot and future pilots the greater the opportunity for identifying reusable artifacts. The GTA needs to also take diversity into consideration; since the state Information technology budget is funded across many agencies the GTA has an obligation not to provide the majority of its services to a minority of constituents. The pilot should be sufficiently broad that a wide range of agencies (constituents) are represented.

Low hanging fruit



A number of critical success factors have been identified for the planning phase of the reuse program. Two of these are highlighted as Milestones; ensuring that the GTA is sufficiently far along in developing its reuse aptitude and that pilot resources are allocated.

The necessity to develop and roll out the reuse program in step with increasing the maturity level of the software development teams cannot be overemphasized. The pilot project team should preferably already be proficient in component based development (CBD) and have a number of successful CBD projects to its credit.

The GTA should adopt an incremental pilot implementation plan, developed during the planning phase. This will ensure that the existing scarce human resources are not over stretched during implementation.

The following resources will be required for the planning and the implementation phase:

Resources	Reuse Champion
	Producer Manager
	Consumer Manager
	Engineers
	Marketing
	Domain Expert
	Domain Analyst
	Asset Manager
	Reuse Analyst
	Metrician

NOTE GTA does not currently have available the following resources with the skills that will be required during the planning phase:

Engineers

Marketing

Domain Expert

Domain Analyst

Asset Manager

Reuse Analyst

Metrician

♦ **Milestone** Creation of a vision statement and design of organization for reuse. Pilot Resources allocated. GTA and target organizations have met agreed reuse aptitude criteria.

4.4 Implementation – reuse pilot program

During the implementation phase a transfer of reuse technology, processes and knowledge occurs from the reuse planning team to the pilot project team.

As the pilot project is implemented the following four factors needs to be taken into consideration:

- Technology Transfer
- Change management
- Implementation mechanism (evolutionary or revolutionary)

- Top down vs bottom up implementation.

The goal of the implementation pilot is that by the end of the pilot the reuse program will have reached a steady state. At the end of the pilot the diffusion of reuse throughout the GTA and changes to the available resources are a key indicator of success.

These indicators will help determine whether previous stages need to be revisited and additional pilot initiatives undertaken in order to reach a steady state of systemic reuse.

Resources	Reuse Champion
	Producer Manager
	Consumer Manager
	Engineers
	Marketing
	Domain Expert
	Domain Analyst
	Asset Manager
	Reuse Analyst
	Metrician

NOTE The GTA does not currently have a pilot project team to transfer knowledge to or the resources to transfer skills from that will be required during the implementation phase.

Accelerated reuse adoption recommendation:

To accelerate the reuse program in the implementation phase the GTA should partner with external development contractors who have existing reuses and CMM level 5 component based development programs.

♦ **Milestone** Adoption of a reuse infrastructure.

4.5 Continuous Improvement

Continuous improvement is the final phase of implementing a reuse program. Critical tracking of systemic reuse metrics occurs. The ongoing resources are the same as those identified during the pilot phase.

Resources See implementation pilot

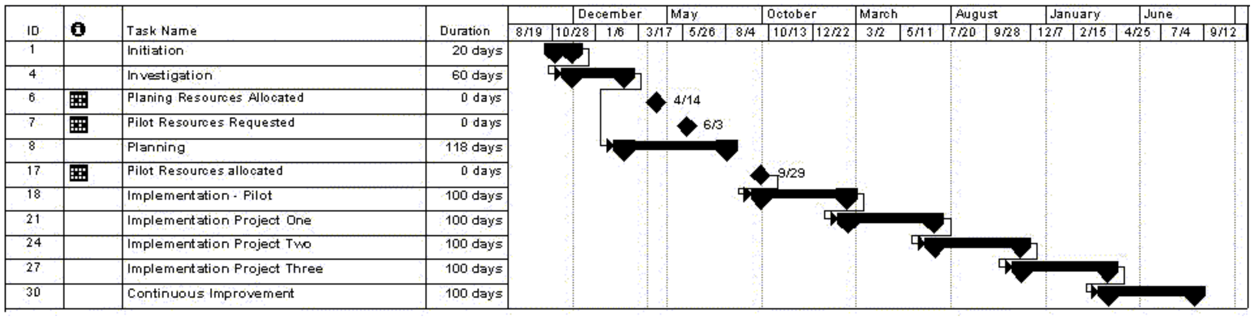
♦ **Milestone** Continually identify problem areas and opportunities for reuse.

4.6 Assumptions and Dependencies

The following assumptions were made:

- Initiation has occurred.
- Investigation has not been undertaken, no evaluation of the GTA / States reuse aptitude has occurred.
- Funding needs to be requested prior to June 2002 to be able to move from planning phase to pilot beginning in fiscal year July 2003.
- Resource need to be identified to move from the investigation phase to the planning phase.
- A secondary parallel initiative to raise the software engineering capability of the GTA to a target CMM level 3 occurs.
- Resources do not exist within the GTA funding to build a target pilot team has not been allocated.
- One or more vendor partners with a mature CBD practice have been identified.

4.7 Time and Resources



ID		Task Name	Duration
1		Initiation	20 days
4		Investigation	60 days
6	▦	Planing Resources Allocated	0 days
7	▦	Pilot Resources Requested	0 days
8		Planning	118 days
17	▦	Pilot Resources allocated	0 days
18		Implementation - Pilot	100 days
21		Implementation Project One	100 days
24		Implementation Project Two	100 days
27		Implementation Project Three	100 days
30		Continuous Improvement	100 days

Task	Duration	Start	Finish
Initiation	20 days	11/1/01	11/28/01
Gather basic information	20 days	11/1/01	11/28/01
Identify Sponsors	5 days	11/1/01	11/7/01
Investigation	60 days	11/29/01	2/20/02
Feasibility Study	60 days	11/29/01	2/20/02
Planning Resources Allocated	♦	4/14/02	4/14/02
Pilot Resources Requested	♦	6/3/02	6/3/02
Planning	120 days	2/21/02	4/16/03
Creation of vision statement	18 days	2/21/02	3/18/02
Implementation strategy	40 days	4/15/02	6/7/02
Define Organizational Structure	40 days	6/10/02	8/2/02
Role & responsibilities	40 days	4/15/02	6/7/02
Develop metrics	40 days	6/10/02	8/2/02
Create marketing strategy	40 days	4/15/02	6/7/02
Create education plan	40 days	6/10/02	8/2/02

Identify pilot	120 days	3/19/02	8/5/02
Pilot Resources allocated	♦	9/29/02	9/29/02
Implementation - Pilot	100 days	9/30/02	2/14/03
Continuous Improvement	400 days	2/17/03	8/27/04

FAQ

Q) What phase of the reuse program are we in?

This document is the vision statement. The vision statement is the deliverable from the planning phase of the reuse program

Q) Why is the planning phase three hundred days in duration?

The resources required to complete the planning and pilot phases have not currently been allocated. The current plan is based on a single dedicated resource working in conjunction with the other resources identified in section 4.3 Planning .

To ensure the GTA has the opportunity to be successful with its reuse program, a number of parallel activities need to occur. Funding needs to be allocated for the pilot projects and a mature software engineering team needs to be built. This timeline provides the GTA with time to accomplish these goals.

Q) Can the program be accelerated?

Yes, it is possible to foreshorten the planning phase. A number of the deliverables described in the planning phase may be undertaken in parallel.

In addition it may be possible to leverage external vendors to develop reusable artifacts.

Q) What resources will be required to implement the program?

Resources are required to implement the program and to sustain it in steady state. To achieve the latter, it is necessary to be able to empower the target organization identified during the pilot projects. These two groups of resources should be treated separately. Some of these resources may be provided by external vendors.

Q) How long will the continuous improvement phase of the program run for?

The two studies identified earlier in the vision document were undertaken over a period of eight and ten years respectively. The length of the programs will vary across different agencies.

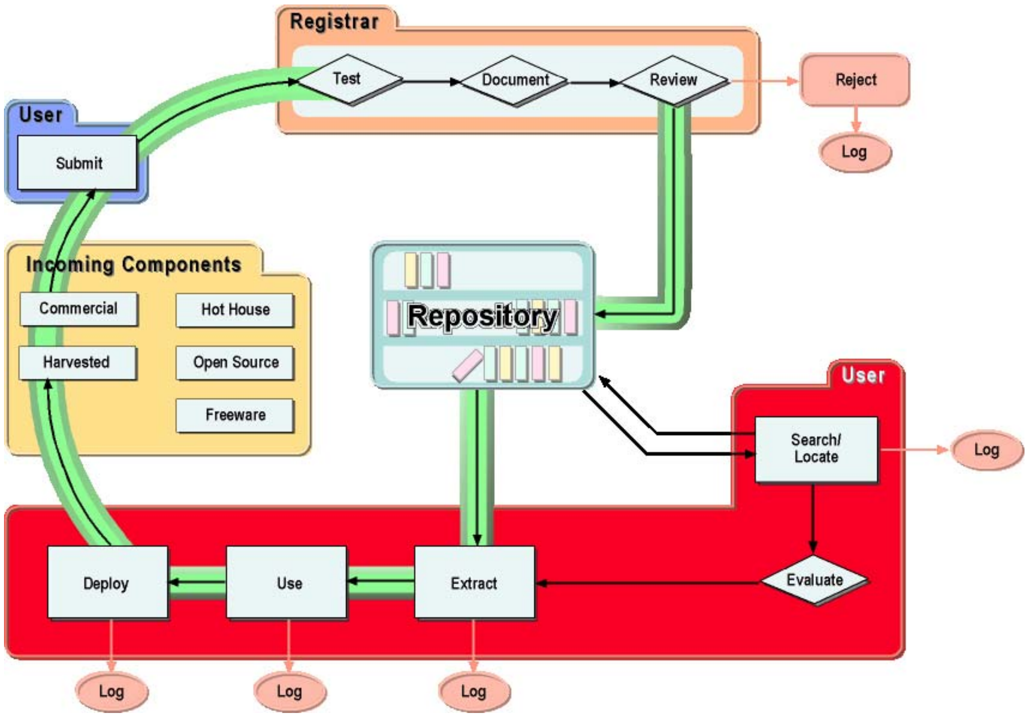
Q) How will this program work with the proposed State component repository market place?

This program will complement the component repository initiative. The reuse team created to support this project will be able to ensure that quality components are commissioned and acquired by the State of Georgia.

5. Program Features

The following list and briefly describe the program features.

5.1 Re-use Artifact Life Cycle



5.2 Vision statement

This document

5.3 Implementation strategy

The implementation or "technology transfer" of reuse is a critical aspect to achieve systemic reuse. A reuse program planned down to the smallest detail is of no value if this technology cannot be successfully transferred to the agencies that would most benefit.

Once suitable candidates for reuse have been identified, the implementation strategy will define the appropriate approach. Multiple approaches will be identified: parallel conversion, direct conversion, phased conversion, pilot conversion, etc.

The strategy will take into account how urgently the GTA wants to transition the organization and the characteristics of the reuse candidates.

5.4 Reuse organizational structure

A reuse organizational structure will be created that facilitates division of labor and coordination of tasks to achieve the common goals of reuse. A wide range of structural forms exist for software reuse; each has its advantages and disadvantages which may be accentuated by particular circumstances in the pilot project.

The reuse functional and project organizations will be defined.

5.5 Reuse roles and responsibilities

Key roles for implementing reuse will be identified. In a small organization these functions can be accomplished by a single person, in large complex organizations an entire team may be required depending on the scope of the selected reuse pilots, their size and the experience levels of the candidate organization.

5.6 Reuse metrics

It's important to establish specific goals for the level of reuse; these goals must be both reasonable and realistic.

What's reasonable and realistic? "It all depends," observes Hewlett Packard Software Technology Laboratory scientist Dr. Martin Griss, "on the domain, the size, distribution, and culture of the organization, the management, [and] the general process maturity."

Cutter Consortium™ consultant and author Paul Harmon offers a similar take. "It depends on the size of the application," he says. "Very simple applications that have been done with slight variations before go much faster. New, complex applications take time."

"I normally set fifteen percent as a goal," reports Lockheed Martin systems analyst Dr. Jeffrey Poulin, author of Measuring Software Reuse. "The more a company wishes to promote reuse through reuse planning and domain-specific libraries, the closer they can get to higher levels, like eighty percent."

Reuse metrics define a way of measuring some attributes of developing software with reusable assets, not just code. The reuse goals of the GTA will be evaluated in order to identify the appropriate critical set of metrics for benchmarks, incentives and decision making.

5.7 Reuse marketing strategy

The essence of reuse marketing is the exchange of assets of value. The marketing concept as applied to reuse is that the key to achieving reuse organization goals lies in identifying the target "candidate" organization's needs and delivering this value through the reuse program.

A reuse marketing strategy will be developed. This may include reuse visionaries, reuse library prototypes, reuse educators, communication plans, product pricing, product launch events, distribution and promotional activities.

5.8 Reuse education and training plan

Education and training are an important means of communication. Often cited reasons for not implementing reuse include:

"We don't have reusability skills"

"I don't believe reuse works"

"Reusable software destroys creativity"

"Reusable software cannot be efficient"

"We don't have a reuse plan"

These can be overcome first by concept training and then by tool training. A core reuse education curriculum will be developed to address these needs.

5.9 Pilot project

Pilot project(s) will be selected and a pilot implementation plan created. The GTA will consider similarity and diversity. The greater the similarity between the business objectives of the pilot and future pilots the greater the opportunity for identifying reusable artifacts. The GTA will also take diversity into consideration.

6. Constraints

7. Other Program Requirements

7.1 Applicable Standards

The following standards have been identified:

Sun Code Conventions for the Java™ Programming Language

http://java.sun.com/docs/codeconv/

This document contains the standard conventions that will be adhered to. It covers filenames, file organization, indentation, comments, declarations, statements, white space, naming conventions, programming practices and includes a code example. The Code Conventions for the Java Programming Language document was revised and updated on April 20, 1999.

The XML Working Group is considering whether to establish a registry of "inherently governmental" data elements, DTDs, and schemas. Wherever applicable this registry of government elements should be used.

Contact Lisa.Carnahan@nist.gov for further information.

Future applications in the State of Georgia will be developed as component-based applications that present a web services interface to the enterprise software infrastructure.

All components created for the state will be constructed to conform to the *Reusable Asset Specification.*

The purpose of the specification is to provide guidelines for the description, development and application of different kinds of reusable software assets. The specification is based on the concepts defined in the Unified Modeling Language (UML), the Rational Unified Process (RUP) and the concepts described in the Architectural Description Standard (ADS).

http://www.rational.com/rda/ras/preview/index.htm

The State of Georgia also recognizes the need for additional standards and component naming conventions.

7.2 System Requirements

N/A

Appendix A Reuse Definitions

Definitions (as required)

For the purposes of this procedure the following terms and definitions apply. IEEE std 1517-1999.[9]

ADS Architectural Description Standard; specifies the views and viewpoints on models and artifacts that are architecturally significant.

ADS View A projection into models and specifications that are architecturally significant.

ADS Viewpoint A collection of views with strong affinity for representing architecturally significant artifacts.

application engineering The process of constructing or refining application systems by reusing assets.

architecture Organization and justification of static and behavioral artifacts for software and its structure.

architecture Description A collection of artifacts documenting the architecture of the system. More precisely, an architectural description is a subset of the system models that best captures and explains the architectural decisions.

artifact An artifact is any item that is input to or output from the software development process, or tools, etc. There may be many artifacts organized in an asset. An artifact may be represented as a file on a file system. An item, such as a design, specification, source code, documentation, test suite, manual procedures etc

articulation The degree to which an asset is described. The level and degree of specification, implementation and testing artifacts describe an asset. Each asset category in the Reusable Asset Specification further describes types of assets for a given profile, possessing varying degrees of articulation.

assemble The process of constructing from parts one or more identified pieces of software

asset An asset is a package of relevant artifacts that provide a solution to a problem. Further, an asset is different from an artifact in that it has several sections which describe it further - namely: overview, classification, solution, and usage. An asset is an asset because it has one or more artifacts and these are classified and have a description of how to apply and use them.

asset profile A grouping of assets that share common scope, variability, granularity, visibility, content, and purpose. For instance, an asset category may be a framework or a mechanism.

asset structure model A model in the Reusable Asset Specification that describes the logical composition of an asset. This model describes the major sections of the asset and the relevant artifacts and descriptors.

asset type A refinement of an Asset Profile, providing more detailed description of the asset or assets. For instance, whereas an asset profile may be "component", an asset type for that profile may be a J2EE component or a .NET component.

black box Also known as verbatim reuse employs existing assets without modification of the software engineering process thereby preserving asset integrity. Under black box reuse the consumer typically sets parameters to

[9] IEEE publication available from the Institute of Electrical and Electronic Engineers (http://www.standards.ieee.org)

instantiate the component and only needs to understand the components interface not its internal function.

bound role This is a role that has a concrete element attached. This can be a class, an operation, any classifier, model element, primitive type or other concrete element. Roles are in the scope and context of a collaboration. See Parameter.

classification The manner in which assets are organized for ease of search and extraction within a reuse library.

collaboration A collaboration is a society of classes, interfaces, and other elements that work together to provide some cooperative behavior that's bigger than the sum of all its parts. A collaboration is also the specification of how an element, is realized by a set of model elements, classifiers, primitive types and associations playing specific roles used in a specific way.

collaboration diagram In general the collaboration diagram is understood by some to represent sequence diagrams or interaction diagrams.

component Refer to the UML 1.3 specification.

component framework Desmond D'Souza puts these concepts together and describes component frameworks this way: "In general, a component framework is a collaboration in which all the components are specified with type models; some of them may come with their own implementations. To use the framework, you plug in components that fulfill the specifications."

component system A Component System is a special kind of Pattern that contains other reusable solutions. A Component System has extension points and parameters to fill and complete. However, unlike a Framework, a Component System cannot execute on its own but rather requires a larger context such as a Framework in which to execute. A Component System must contain at least one Mechanism.

construction The process of writing, assembling, or generating assets

descriptors These are descriptors for classifying an asset. These descriptors may be used for searching for assets.

domain A problem space

domain analysis The analysis of systems within a domain to discover commonalities and differences among them.

domain architecture A generic, organizational structure or design for software systems in a system.

domain model A Domain Model describes the business context in which the system will operate. This includes the business objects, concepts, and integrity rules of the domain. The stakeholders of this asset include designers, developers, and users. The Reusable Asset Specification uses a collection of domain models to describe the Specification and the core semantics of assets.

domain definition The process of determining the scope and bopundaries of a domain

domain engineer A party that performs domain engineering activities

domain engineering A reuse based approach to defining the scope , specifying the structure and building the assets

domain expert An individual who is intimately familiar with the Domain and can provide detailed information to the domain engineer domain model

external view This refers to the collaboration diagram and icon showing the parameters of a specific collaboration.

framework A Framework provides an extensible template for applications within a domain. A Framework is bigger than a Mechanism. It is a micro-architecture for a system, encompassing a set of collaborations, model elements, and classifiers that work together to solve a common problem for a domain.

The UML defines Framework as: 1) A stereotyped package consisting mainly of patterns. 2) An architectural pattern that provides an extensible template for applications within a specific domain.

generalizable element Refer to the UML 1.3 specification.

grey box Reuse occurs when the reuse component is not modified , but a variant is deployed by a specialization technique that creates a new component. An example is a in OOP where a sub class inherits the characteristics of a super class.

internal view These are the internals of a collaboration such as the class diagrams and sequence diagrams. Typically it is within this view that one can see where to extend and fill the parameters.

mechanism A Mechanism is a special kind of Pattern. A mechanism is a design pattern that applies to a society of classes.

mining (aka harveting) The activity of extracting reusable content, assets and architectures.

model A representation of a system at a chosen level of abstraction created in order to help understand the system's structure and operations

namespace Refer to the UML 1.3 specification.

non-functional requirement Non-functional requirements capture architecturally significant, non-business requirements and constraints.

parameter A parameter is an extension point that is viewed on a collaboration. The parameter can have type and default value information as part of its description.

parameterized collaboration This is a collaboration that has one or more parameters.

pattern A pattern is a common solution to a common problem in a given context. [1] For the purposes of this Specification, we intend pattern to mean a design pattern. A pattern is the most flexible in terms of manipulating its participants. By definition we say that all participants in a pattern are parameterized.

pattern language A Pattern Language is a collection of interrelated patterns that combine to solve a problem and may be viewed as a single Pattern together. Specifically these contained patterns are Templates, each of which share some of the same context as defined by the Pattern Language.

pattern system A Pattern System is an abstract class in the Content meta model that is a special kind of Pattern that contains other reusable solutions. A Pattern System is also a solution to a recurring problem in a given context – but it can contain other such solutions.

problem description A concise description containing the reasons and requirements for which the asset has been developed.

product line A collection of systems that are potentially derived from a single domain architecture

RAS Reusable Asset Specification

realization A Realization is a special kind of Pattern, providing a common solution to a common problem in a given context. The distinguishing characteristic of Realizations is that they have no unbound roles. All roles on Realizations are bound, meaning filled in with concrete elements. This means that the consumer of this type of asset does not alter nor customize the asset.

reusability The degree to which an asset can be used in one or more software systems or in building other assets (see reuse metrics)

reuse The use of an asset in the
reuse sponsor

reusable (Software) asset A reusable software asset is a software artifact or a set of related artifacts that has been crated or harvested with an explicit purpose of applying it (repeatedly) in a subsequent, separate development efforts. Reusable software assets can be of different granularity, may allow different degrees of customization (or variability) and can be applied (or targeted) at different phases of software development.

reuse guidelines How the consumer should approach and apply an asset.

reuse metrics A collection of metrics describing qualities of an asset such as how often it is used, its version, etc.

role This is the behavior and characteristics that a given model element, classifier or other element can take on for a given period of time.

root context A package from the UML that is defined in the Reusable Asset Specification for containing the artifacts of a reusable asset.

RUP Rational Unified Process

significant architectural requirements (SARs) Those functional requirements, system properties and/or constraints that influence significant architectural decisions. Those requirements that impact architecture the most are grouped together into a set called SARs.

software system A set of run-time software components and computing devices (on which these components are deployed) that interoperate to provide the system's functionality.

solution The Solution section of an asset is a collection of artifacts providing a solution to a problem. These artifacts are in two sub-sections, namely: Specification and Implementation.

stakeholder A person or role that has interest in and influence on the system

system An integrated composite that consists of one or more processes

systematic reuse The practice of reuse according to a well defined repeatable process.

template A Template is a special kind of Pattern, providing a common solution to a common problem in a given context. A Template is the most flexible in terms of manipulating its participants, as compared to mechanisms, frameworks, and component systems. By definition we say that all roles in a Template are parameterized, and therefore can be customized.

Templates provide a critical element to capturing reusable assets and architectures by documenting the justification and reasoning for design decisions. This is a quality that allows assets to survive to the next generation of developers and designers.

unbound role A role in a collaboration that does not have a concrete element specified.

UML Unified Modeling Language

usability A measure of an executing software unit's or systems functionality, ease of use and efficiency

usage A collection of artifacts describing the core extension points and techniques for using an asset. This section of an asset provides key insights to making the asset approachable and consumable.

use case A use case is a description of a set of sequences of actions, including variants, that a system performs to yield an observable result of value to an actor.

variability point A location in an asset that may be altered, customized, modified, or supplied by an asset consumer. A variability point can define the assets variability at design time, code time, and run time.

view A projection (subset) of the system models that shows a specific aspect of the system or addresses one or more of the concerns of the system stakeholders

whitebox Also known as leveraged reuse, is by far the most common type of reuse and begins with existing assets in the software development process modifying them as needed to meet specific requirements.

Appendix B Bibliography

IEEE Standard for Information Technology – Software Life Cycle Processes – Reuse Processes, IEEE-std 1517-1999

R.Prieto-Diaz, “Making software reuse work: an implementation model,” SIGSOFT Software Engineering Notes, vol.16 July 1991

B. Whittle, W L.Lam, and T.Kelly , “A pragmatic approach to reuse introduction in an industrial setting” 1996

Wayne C. Lim “Managing Software Reuse, A comprehensive guide to strategically re-engineering the organization for reusable components” Prentice Hall 1998

J.S. Poulin “Measuring Software Reuse, Principals, Practices and Economic Models” Addison Wesley 1997

D. J Reifer “Practical Software Reuse, Strategies for introducing reuse concepts in your organization” John Wiley & Sons 1997